

## Nomi e convenzioni

### **Form**

upload.jsp

### **Struts form**

uploadForm.java

### **Struts action**

uploadAction.java

### **Hbm della tabella**

file.hbm.xml

### **Classe della tabella**

file.java

## **1. Creazione Tabella SQL**

Prima di tutto è necessario creare una tabella in grado di memorizzare un file all'interno di un database SQL.

```
CREATE TABLE 'file' (  
  'ID_file' int(10) unsigned NOT NULL auto_increment,  
  'attach' longblob,  
  `attach_size` int(10) unsigned default NULL,  
  `attach_content` varchar(45) default NULL,  
  `attach_name` varchar(45) default NULL,  
  PRIMARY KEY (`ID_file`),  
)
```

## **2. Configurazione di Hibernate**

A questo punto bisogna creare il file di Mapping e la classe Java di Hibernate, il tutto si può semplificare utilizzando MyEclipse per effettuare l'operazione in automatico. Ad ogni modo il risultato dovrà contenere

### **file.hbm.xml**

```
<id name="idFile type="integer">  
  <column name="ID_file" />  
  <generator class="increment" />  
</id>  
<property name="attach" type="blob">  
  <column name="attach" />  
</property>  
<property name="attach_size" type="integer">  
  <column name="attach_size" />  
</property>  
<property name="attach_content" type="string">  
  <column name="attach_content" />  
</property>  
<property name="attach_name" type="string">  
  <column name="attach_name" />  
</property>
```

**file.java**

```
import java.sql.Blob;
.....
private Integer idFile;
private Blob attach;
private Integer attach_size;
private String attach_content;
private String attach_name;
.....
```

**file.hbm.xml**

```
<id name="idFile type="integer">
<column name="ID_file" />
<generator class="increment" />
</id>
<property name="attach" type="blob">
<column name="attach" />
</property>
<property name="attach_size" type="integer">
<column name="attach_size" />
</property>
<property name="attach_content" type="string">
<column name="attach_content" />
</property>
<property name="attach_name" type="string">
<column name="attach_name" />
</property>
```

**file.java**

```
import java.sql.Blob;
.....
private Integer idFile;
private Blob attach;
private Integer attach_size;
private String attach_content;
private String attach_name;
.....
```

**3. Inserire nel file struts-config.xml le informazioni relative al nuovo Form**

```
<form-beans>
<form-bean name="uploadForm" type="percorso del file uploadForm.class" />
</form-beans>
.....
<action-mappings>
<action
attribute="uploadForm"
input="percorso del file upload.jsp"
name="uploadForm"
```

```

path="/upload"
scope="request"
type="percorso della classe uploadAction.class">
</action>
</action-mappings>

```

#### 4. Struts Form

##### uploadForm.java

```

import org.apache.struts.upload.FormFile;
.....
private FormFile fileUpload;
.....
public void reset(ActionMapping mapping, HttpServletRequest request) {
fileUpload = null;
}

```

#### 5. Fom JSP

##### upload.jsp

La pagina in questione sarà l'unica che verrà visualizzata all'utente finale

```

.....
<html:form action="/upload" enctype="multipart/form-data">
<html:file name="uploadForm" property="fileUpload"/>
</html:form>
.....

```

#### 6. Azione attivata dal Form

##### uploadAction.java

```

public ActionForward execute(
ActionMapping mapping, ActionForm form,
HttpServletRequest request,
HttpServletResponse response) throws FileNotFoundException, IOException {
uploadForm action = (uploadForm) form;
// Apertura connessione con Hibernate
.....
file f = new file();
f.setAttach(Hibernate.createBlob (action.getFileUpload.getInputStream()));
f.setAttach_size(action.getFileSize());
f.setAttach_name(action.getFileName());
f.setAttach_content(action.getContentType());
....
}

```